Final

Dalton Murray

College of Business & I.T., Lawrence Technological University

INT 4203, Systems Analysis and Design

Dr. Anthony Padalino

December 13, 2022

**Final**

**1. Why would a systems analyst have to act as a translator? What groups might be involved?**

Systems analysts have to act as translators so that when they explain a system, or processes to programmers or to management they can explain what needs to happen in the system, and they have to do it in a way that the person they are speaking to is able to understand what happens, which is why they must act as a translator. In the book, *Systems Analysis and Design*, it explains “when they describe business processes to programmers, they must speak a language that programmers will understand clearly. Typically, the analyst builds a series of models, diagrams, and decision tables and uses other descriptive tools and techniques. Similarly, when communicating with managers, the analyst often must translate complex technical issues into words and images that nontechnical people can grasp. To do this, the analyst uses various presentation skills, models, and communication methods” (Tilley, S. p. 28 - 29). In other words, this means that an analyst must adapt to who they are talking to so that they can understand the system, and they must do this in a few different ways like using models, diagrams, decision tables, presentations, and verbally. When a system analyst speaks to a programmer about a system, they are more likely to speak more technically, when a system analyst speaks to a manager, they are more likely to speak less technically in order to explain the system. It is also possible that they communicate about completely different topics, with a programmer they may talk about exactly how the system runs, but with a manager they may say why they benefit from the system and not how it runs exactly.

When a system analyst acts as a translator, they must involve a few different groups. Programmers and designers of the system will likely be involved, as well as supervisors, and different levels of management.

**2. What is project scope? What are constraints? Provide an example of a mandatory, external, future constraint. Also provide an example of a discretionary, internal, present constraint.**

A project scope is what gives a project the boundaries of the project, what is to be done in the project, as well it should include what is exclusively out of the scope. This way, if a change request occurs, it is possible for it to be accepted if it’s in the scope of the project, however, if it’s outside of the scope of the project it is more than likely to get rejected. However, there are, of course, cases where a change request gets accepted even if it’s out-of-scope. The book defines a project scope as “defining the specific boundaries, or extent, of the project. […] Some analysts find it helpful to define project scope by creating a list with sections called Must Do, Should Do, Could Do, and Won’t Do” (Tilley, S. p. 62). In other words, the project scope says what is exactly expected of the project. It is also a good idea with a project to explain what are things that must be done to be considered completed with the project, what should should be done, however, might not be done, what things could be done, but likely will not be done, and things that will not be done due to a few different reasons like outside of budget, or just not feasible in other ways. It is also important when conducting a project to be wary of scope creep, scope creep often occurs when the project scope was not defined correctly, or change requests were done which were not supposed to be apart of the project, or by bad communication. Scope creep means that the project requirements may be growing without specific authorization.

A constraint on a project is “a requirement or condition that the system must satisfy or an outcome that the system must achieve […] can involve hardware, software, time, policy, law, or cost. System constraints also define project scope” (Tilley, S. p. 63). In other words, a constraint is something that is known on a project that may be a requirement for the system or project to be completed in order for it to be a successful project. An example of a mandatory constraint is if management tells you that the project must be done on existing hardware or be an addon to existing software. An example of an external constraint is if a government agency has requirements on the system, for example it must store data for a certain amount of time. An example of a future constraint is if the client tells you that in the future, they require something to be done. An example of a desirable constraint is if we want to reduce the cost of operating a system, we should scale it down, while a discretionary project is where choices exist, so we could take the same thing for constraints and a discretionary constraint example could be we can either place an order for a certain amount of product for the future because the price is down now, or we could wait. An example of an internal constraint is if the company places a constraint on itself, such as telling programmers of the website that they will switch over to using 2FA. An example of a present constraint is if management says that the project must have certain requirements now, rather than later, like it must have some function now and more can be added later.

**3. What is a task? What is an event? What is a milestone?**

A task is also known as an activity and is “any work that has a beginning and an end and requires the use of company resources such as people, time, or money” (Tilley, S. p. 78). In other words, a task is something that must be done in a project which has a clear beginning and end, it also must use company resources. An example of a task could be something simple such as creating a new button on a form, it has a clear beginning and end, and it requires the usage of people and their time. However, it is important to note that a task is not an event or milestone.

An event is also known as a milestone, and it is used as a reference point in a project which is used to monitor progress (Tilley, S. p. 78). Events, or milestones, are often not tasks, however, mark the completion of certain points of time which involves tasks, such as completion of the foundation of the building, which could have had a large number of tasks in it. Another example of a milestone could be getting management approval for something or could be marking the completion of different phases of a project.

**4. What is a systems requirement, and how are systems requirements classified?**

A systems requirement is something like a characteristic or feature which must be included in the system in order to meet requirements and be a successful project which satisfies user’s needs (Tilley, S. p.105). Systems requirements also “server as benchmarks to measure the overall acceptability of the finished system” (Tilley, S. p. 105). Systems requirements also often the focus of the start of the analysis phase, and it is extremely important to know ahead of time all of the requirements for the system so that in development it can be successful, otherwise if there are missing features, the project may not be considered a success, or even in some cases not useful anymore if requirements are missing. It is important with systems requirements to use something called requirements engineering, which has three main activities, understanding the problem, describing the problem, and agreeing upon the problem (Tilley, S. p. 105).

Systems requirements are classified by two ways, a functional requirement, and a non-functional requirement. In order for a requirement to be a functional requirement it must be a statement of the services in which a system provides, such as a website needing to generate reports for volume statistics hourly, or other reports like sales (Tilley, S. p. 105). A non-functional requirement, or quality attribute, is a statement in regards to its operational system constraints, such the data in a report must be uniform with other reports, or it must support a certain amount of users concurrently (Tilley, S. p. 105).

**5. Describe data and process modeling concepts and tools.**

With data modeling, what you are doing is visualizing the flow of the data throughout a project or system. With process modeling, it is the idea of visualizing the flow of the processes throughout a project or system. It is also possible to do both without a diagram in order to visualize the flow of data or processes and can be written out to explain the flow by words instead of by diagram. Data modeling and process modeling are often used in order to help reduce complexity of the system to make it more easily understandable by anyone, and to assist in developing systems or in projects.

With data modeling, there are Data Flow Diagrams, or DFD, and it is a tool used to be able to model data/to be able to visualize the flow of data. It shows how a system is able to store, process, and transform data (Tilley, S. p. 131). When using data flow diagrams, it depends on the level you are making the diagrams, for example if there is a box, which is a process, you may not need worry about what is inside the individual process and only the data input and output at a higher level. However, with lower depth/levels you may want to expand on these individual processes and detail them out what exactly is being done (Tilley, S. p. 131). Some data flow diagram examples can be found below. There are also rules for diagramming such as avoiding spontaneous generation, black holes, and gray holes (Tilley, S. p. 149).

**Diagram

Description automatically generated**

(Tilley, S. p. 159).

A Business Process Model, or BPM, is a tool used to be able to model processes/visualize processes in a system or outside of a system like the processes in a store. With a BPM it shows the people, events, and interactions. However, there are also lots of other ways to create process models. An example of a business process model may be found below.

Diagram

Description automatically generated

(Tilley, S. p. 10).

Both Data Flow Diagrams and Business Process Models are also able to be made with another tool, called a CASE tool or by hand if needed (Tilley, S. p. 129). There is also different software’s which are able to be used to create diagrams such as Visio, Excel, and DrawIO, or just about anything which is able to create any sort of diagram can be used to create these types of diagrams. These diagrams are also able to have differing levels of depth of visualization such as a 40,000-foot view and a 10,000-foot view and could be classified into different levels. For example, a level 0 may be a 40,000-foot view and a level 1 may be a 30,000-foot view, and the higher level the lower level it becomes. When diagramming there are different ‘languages’ or styles used, often these ones done by using UML, or Unified Modeling Language, which uses object-oriented concepts and can have different sets of symbols for different meanings like the Gane and Sarson Symbols or the Yourdon Symbols (Tilley, S. p. 146).

**6. What is object-oriented analysis, and what are some advantages of this method?**

Object-oriented analysis, also called O-O analysis, is defined as “views the system in terms of objects that combine data and processes. The objects represent actual people, things, transactions, and events. Compared to structural analysis, O-O phases tend to be more interactive. Can use the waterfall model or a model that stresses greater iteration.” (Tilley, S. p. 18). Simplified, object-oriented analysis is an approach we are able to use in how we see a system, how we design and develop a system, and how we interact with a system. Object-oriented analysis is used to focus on using objects rather than seeing the system as data, and processes. There is also a thing called object-oriented diagramming, which is used to help create the system as well as business process modelling is used to help create the system. These two diagramming types allow us to be able to understand the system better so when in development it can be more easily made.

According to our book, some advantages of using object-oriented analysis are that it “integrates easily with object-oriented programming languages. Code is modular and reusable, which can reduce cost and development time. Easy to maintain and expand because new objects can be created using inherited properties” (Tilley, S. p. 18). This means that some advantages of using object-oriented analysis over other methods of viewing systems, such as structured analysis, is that it integrates with object-oriented focused programming languages such as Python. This allows a programmer to be able to be more productive and efficient in design. It is also modular and reusable, allowing you to make one object in the program and then call back to it with new inputs and get new data using the same functions rather than programming a new object which is going to do the same thing. Doing this reduces the amount of time the programmer must put into the program. It is also easier to maintain and more simplistic than other methods. Moreover, object-oriented analysis may easily integrate with the Unified Modeling Language, UML, so that we can document and visualize an information system more easily (Tilley, S. p. 181).

You can also see the main differences between object-oriented analysis and structured analysis, along with different types of example diagrams in the below 4 images. The first image shows a structured-analysis diagram which focuses on the flow of the data and processes, the image below shows an object-oriented diagram which shows the interaction of the objects, below that is the process of doing structured-analysis which uses a waterfall model and below that is the object-oriented process flow which allows tasks to occur simultaneously.

Diagram

Description automatically generated

Structured-analysis example diagram

(Tilley, S. p. 19).

Diagram

Description automatically generated

Object-oriented analysis example diagram

(Tilley, S. p. 22).

Diagram

Description automatically generated

Structured analysis flow

(Tilley, S. p. 20).

**Diagram

Description automatically generated**

Object-oriented flow

(Tilley, S. p. 22).

**7. What are three typical reasons why companies develop their own information systems?**

There are many different reasons why a company may choose to develop their own information system. These reasons include: satisfying unique business requirements, minimizing changes in business procedures and policies, meeting constraints of existing systems, meeting constraints of existing technology, and developing internal resources and capabilities (Tilley, S. p. 205).

Satisfying unique business requirements – It is possible that there are no existing available commercial packages for software which needs to be used, and if you have a business with unique requirements, the system may need to be made specifically for the company (Tilley, S. p. 205). It is possible; however, an existing system exists, but just doesn’t meet the requirements and everything needed (Tilley, S. p. 205).

Minimizing changes in business procedures and policies – When looking for a system it is possible that the new system would require the business to change procedures and policies, however, if they developed their own system they would not need to do this (Tilley, S. p. 205).

Meeting constraints of existing systems – If there are existing systems which the new system has to be able to integrate with, it is possible that it will not work properly or at all, so developing a new system in-house can guarantee that the system will work (Tilley, S. p. 205).

Meeting constraints of existing technology – It is also possible that a new system requires new hardware, and if the company does not want to purchase new hardware they can make a system to be able to work on their hardware (Tilley, S. p. 205).

Developing internal resources and capabilities – If you develop the system in-house it ensures that your programmers are trained on it and can train others on it and don’t need to pay to get in people from a third party, it also means that they are able to make changes to it in the future and fix problems if they arise instead of paying the third party to do all of this (Tilley, S. p. 205).

**8. Describe the habits of successful interface designers.**

There are seven habits of successful interface designers, these habits are as follows: understanding the business, maximizing graphical effectiveness, think like a user, use models and prototypes, focus on usability, invite feedback, and document everything (Tilley, S. p. 232 - 234).

Understanding the business – It is very important for the designer to understand the business and its functions, this lets them know how they need to make the system so that it is able to support individual, departmental, and enterprise goals, allowing users to perform their jobs properly (Tilley, S. p. 232).

Maximizing graphical effectiveness – It’s important for designers to be able to maximize the effectiveness of a system graphically, as that is how the user experiences the system, making sure that the system is able to be learned better through visual usage allows the user to learn the system faster, it also allows users to be more productive (Tilley, S. p. 232).

Think like a user – When designing a system it’s important to be able to think as a user, what they know, how they are going to experience the system, as well as recognize the skill level required to be able to use the system, allowing the designer to be able to make the system easier to learn (Tilley, S. p. 233).

Use models and prototypes – The interface of a system for a user is the most critical part, it is how they will interact with the system, and it is extremely important to make models and prototypes of the system and then allow users to review it, provide feedback, and approve of it (Tilley, S. p. 233).

Focus on usability – When designing a system it’s important to focus on how it is going to be used, and what can be accessed through the interface like tasks, commands, and communication between users and the system, as well as it’s important to figure out where and how you are going to display all interactions, and make sure not to confuse the user with too much information on a single page (Tilley, S. p. 233).

Invite feedback – One of the most important things as a designer is to be able to invite feedback, this allows you to make the system and experience much better than it would have been without having feedback (Tilley, S. p. 233). It’s important to get feedback from users, managers, as well as anyone else who is going to use the system in any capacity, such as customers or third party vendors (Tilley, S. p. 233). It’s also a good idea to observe how a user uses the system to see what they may struggle on (Tilley, S. p. 233).

Document everything – When designing a system documentation is very important, every design should be documented for use later on by programmers, allowing them to program functionality and know what everything should do, as well it allows for later on when you might’ve forgot something to be able to go back and look at it (Tilley, S. p. 234). Documentation in design is also important because one day you may no longer work at the company, and future employees need to know the information too (Tilley, S. p. 234).

**9. What are data warehousing and data mining? How do businesses use these tools?**

A data warehouse is a “integrated collection of data that can include seemingly unrelated information, no matter where it is stored in the company. Because it can link various information systems and databases, a data warehouse provides an enterprise-wide view to support management analysis and decision making” (Tilley, S. p. 301).In other words, a data warehouse typically stores a large amount of information, whether it is related or not to each other, and regardless of where it is stored in the company and can be linked to various different systems and databases. There are also things called a data mart, however, these are designed specifically to serve the needs of a department like sales, rather than every department in the whole company (Tilley, S. p. 301).

Businesses use data warehouses as they allow people to see a larger picture in order to support management analysis and decision making, it also allows the user to specify certain dimensions or characteristics (Tilley, S. p. 301). For example, if a manager wanted to see a sales results for a specific month and year they can see all of this, and then can specify it down further to see what more exactly they want, in order to support decision making (Tilley, S. p. 301).

Data mining is software which is used to look for “meaningful data patterns and relationships. For example, data mining software could help a consumer products firm identify potential customers based on their prior purchases” (Tilley, S. p. 302). Data mining has also brought up concerns about ethical and privacy, and how businesses are able to use, buy and sell data which they get (Tilley, S. p. 302). A large portion of data mining is used as a marketing tool with the goal of increasing the number of pages viewed, increase number of referred customers, reduces clicks to close, increase checkouts per visit, and increase average profit per checkout (Tilley, S. p. 302). When used in these cases, it is also sometimes called clickstream storage (Tilley, S. p. 302). Data mining is also useful to be able to build profiles of typical new customers, as well as returning customers, and those who browse but never buy any products (Tilley, S. p. 302).

Data mining is used as a business tool to make a large amount of decisions based on customers, for example who to send out coupons to, to see trends on products and types of customers, as well as figuring out the optimal placement of items (Tilley, S. p. 302).

**10. Describe four types of system maintenance and provide two examples of each type.**

Four types of system maintenance are corrective maintenance, adaptive maintenance, perfective maintenance, and preventative maintenance (Tilley, S. p. 404).

Corrective maintenance – Corrective maintenance is conducted when a system has errors or other faults, and during corrective maintenance it fixes these (Tilley, S. p. 404). However, during corrective maintenance you will also want to diagnose and correct other errors if found.

Example 1 – My first example of corrective maintenance is if a network switch fails, then we will want to diagnose this issue and correct it by either replacing or repairing the network switch.

Example 2 – My second example of corrective maintenance is that a bug was found on a system and the systems start to immediately fail, we will quickly implement a patch in order to keep the system going before we can then properly diagnose and correct the bug in the system.

Adaptive maintenance – Adaptive maintenance consists of adding enhancements, new features or capabilities, to an operational system which has the goal of making the system easier to use (Tilley, S. p. 406). Usually, adaptive maintenance stems from a changing business environment like new products or services (Tilley, S. p. 406). It is also important to identify if the adaptive maintenance is either major or minor, and then from there the process is different (Tilley, S. p. 406).

Example 1 – An example of performing adaptive maintenance is if you want to add support for a new system such as adding 2FA.

Example 2 – Another example of performing adaptive maintenance is adding a new reports screen for different levels of employees like a supervisor or manager.

Perfective maintenance – With perfective maintenance, the goal is to make a system more efficient, reliable, or maintainable (Tilley, S. p. 406). With perfective maintenance, IT usually initiates the maintenance, while with corrective or adaptive maintenance the user initiates the change (Tilley, S. p. 406). When performing perfective maintenance, a result of it is making a system easier to maintain, however, this usually results in a less costly and less risky system (Tilley, S. p. 406). It is also possible to make a complex system less complex with new technology, resulting in easier maintenance (Tilley, S. p. 406). Perfective maintenance is also commonly added on top of other maintenances, such as a new function being added to the system, and then performing perfective maintenance on the whole system (Tilley, S. p. 406). Analysts will also perform software reengineering during perfective maintenance, allowing them to identify potential quality and performance improvements (Tilley, S. p. 406).

Example 1 – The first example of perfective maintenance is performing optimization on databases, resulting in potentially better reliability, efficiency, and maintainability

Example 2 – The second example of perfective maintenance is upgrading or replacing old out of date hardware.

Preventive maintenance – With preventive maintenance, the goal is to look at and identify areas which troubles are likely to occur and then fix it (Tilley, S. p. 407). Like perfective maintenance, IT usually initiates the maintenance (Tilley, S. p. 407).

Example 1 – One example of preventive maintenance is upgrading or installing new antivirus and firewall software and hardware.

Example 2 – Another example of preventive maintenance is making backups and regularly testing them to ensure that they can work.

**11. Reflect back on this semester. What are your key learnings in this term? (1 word to 1 page)**

Throughout the semester I have learned a few different things about systems as well as their design. However, one of the largest things I developed in this semester is the ability to make different types of diagrams in a better way than which I was able to do before. I believe one of the most important things which I learned in this semester was all of the different available tools we have to use in order to develop and design systems, like CASE tools, as well as all of the different tools we talked about in class. I believe one of the most important pieces of knowledge I gained in this semester was what actually a systems analyst does, and how they are used to connect programmers, managers, and clients together. I also liked when we learned about systems analysts how they use different fact-finding techniques to be able to find out what they need to know, as well as if it is quantitatively or qualitatively.

References

Tilley, S. (2020). Systems analysis and design (12th ed.). Cengage.

I pledge that on all academic work that I submit, I will neither give nor receive unauthorized aid, nor will I present another person's work as my own.

Dalton Murray